Ecole Nationale Supérieure des Mines
SAINT-ETIENNE

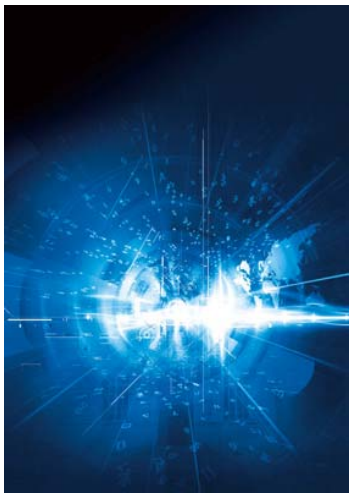
Industrial Engineering and Computer Sciences Division (G2I)

DEALING WITH XML STRUCTURE THROUGH "READING CONTEXTS"

X. TANNIER

Avril 2005

RESEARCH REPORT
2005-400-007



MINISTÈRE DE L'ÉCONOMIE
DES FINANCES ET DE L'INDUSTRIE

Dealing with XML structure through "Reading Contexts"

Xavier Tannier
tannier@emse.fr

Abstract

Some tags used in XML documents create arbitrary breaks in the natural flow of the text. This may constitute an impediment to the application of some methods of information retrieval. This article goes back over an existing tag categorization allowing to distinguish different ways to manage textual content of XML elements. It gives for tag classes a clear definition, through the introduction of a new concept of "reading contexts". Furthermore it proposes a method that uses natural language processing techniques in order to find the class of XML tag names automatically. This work notably allows to recognize emphasis tags in a text, to define a new concept of term proximity in structured documents, to improve indexing techniques, but also to open up the way to advanced linguistic analyses of XML corpora.

1 Introduction

XML (eXtensible Markup Language [18]) is more and more widely used to store and exchange information. XML documents provide textual information with some structural and semantic annotations represented by element tags and attributes.

The following is some useful and basic terminologic indications that are important to understand the article:

(1) <example>My first XML element</example>

In this example:

- "example" is the *tag name*;
- `<example>` and `</example>` are *tags* (respectively start and end tags);
- start/end tags and their content (between them) constitute an *element*.
- Finally, the Document Type Definition (DTD) is a document defining the elements that can be used in the XML file as well as their structure (imbrication, number, sequences, etc.).

Here we focus on a *document-centric* view on XML documents. In this view, mark-up serves for giving information about logical structure and/or about form of a traditional document. This is the case of all texts intended for human people, such as manuals, books, articles or static web pages. This view is opposed to *data-centric* view, used for more database-oriented applications (flight schedules, catalogues, etc.).

Information Retrieval for XML is confronted with a particular problem: on the one hand, document structure is described by human experts in a meaningful and flexible way. On the other hand, for any XML processor, tags are all equally and totally meaningless. Furthermore, names used for semantically equivalent tags can have big variations from corpus to corpus (with different DTDs). Finally they are rarely "real" words, but more or less obscure abbreviations instead. In spite of the efforts of some related projects (XML Schema [20], Semantic Web [19], ...), many document collections lack and will lack serious semantic metadata.
This does not only raise "top-level" problems (as semantic relations between tags or information extraction), but also, as we will see, more "basic" issues such as original text preserving or indexing.

A simple and intuitive division between three classes of tags has been proposed [8], that eases text-searching in XML documents. Section 2 of this article cites other researchs in this field and describes this classification; section 3 proposes a method for determining the category of a tag name in a corpus. Some experimentations are then described, and section 5 discusses the different uses that can be found and the prospects opened by our work, among which: a new definition of term proximity; an automatic recognition of emphasis tags; a method for easing indexing; and a framework for natural linguistic processing of semi-structured data.

2 Three types of XML markup

Bringing some semantics to documents through the markup categorization is not a new idea [16, 12]. In 1987, *Coombs et al.* [4] distinguished six kinds of markup, and lay stress on three of them: "Presentational" markup describes the visual appearance of the text. "Procedural" markup gives instructions to a text formatter, in order to create human-readable presentational markup. Finally, "descriptive" markup gives a label, a role to fragments of text.
Afterwards, "procedural" (formatting instructions) vs. "descriptive" (logical components, like paragraphs, chapters or titles) distinction, which can be considered as a generalisation of the HTML "logical vs. physical" dichotomy, has become quite consensual within the framework of SGML [7], and later XML.
Other thoughts have been given ever since (recently [11, 10, 12, 15]), leading to a bunch of new concepts. Among these concepts we can cite a two-axis discrimation (logical and renditional domains on the one hand, imperative and indicative mood on the other hand) [11] or the proleptic and metaleptic markups [10].

An Information Retrieval oriented division of tags has been proposed by [8], in order to identify different categories that it would be important to distinguish within the framework of XML document retrieval. The original idea was to allow different treatments while searching for a pattern (sequence of characters). We will see (section 5) that, in our opinion, the advantages go far beyond this. The rest of this section gives some details and comments about this last categorization.

2.1 Hard, soft and jump tags

The three different classes are the following:

- *"Hard" tags* are the most frequent, they interrupt the "linearity" of a text, they generally contribute to the structuring of the document. Examples of this type are titles, chapters, paragraphs. Tags `'news'` and `'item'` are both "hard" in the following example:

  (2) <news>
  <item>A new study about evolution of tourism in the United States</item>
  <item>Elections in Ukraine: the Central Commission has published the official results</item>
  </news>

- *"Soft" tags* identify significant parts of a text, like quotations, appearance effects, but become "transparent" while reading the text. This is the case of tags `'bold'`, `'italics'`, `'underlined'` and `'sc'` (small capitals) in examples 3.a, 3.b, and 3.c.

  (3) a. <par>
  <bold>United States elections</bold>
  are administered at the state and local levels.
  </par>

b. <title>
Noam Chomski's comments about
<italics>United States</italics>
<underlined>elections</underlined>.
</title>

c. <title>
U<sc>nited</sc> S<sc>tates</sc>
E<sc>lections</sc>.
</title>

- *"Jump" tags* are used to represent particular elements, like margin notes, references to bibliography, or glosses. They are detached from the surrounding text. Elements `'comment'` and `'footnote'` in the following examples are "jump" elements.

(4) a. <oral_transcription>
I heard the news today about United
States elec<comment>a door
snaps</comment>tions.
</oral_transcription>

b. <paragraph>
The 2004 United States<footnote>See
an article about the United States of
America on page 142</footnote> elections caused less controversy than in 2000.
</paragraph>

c. <abstract>
This document deals with 1995 and 2002 Jacques Chirac <footnote>J. Chirac, the french president, is, by the way, not a really good friend of the president of the United States</footnote> elections.
</abstract>

2.2 Comments

2.2.1 Exceptions

Some elements, such as tables, lists or mathematical environments, which are not treated by the authors, appear to be specific. As a first approximation, we can consider the first two as hard elements, but their particular features may require more careful thoughts (see for example [2] for lists and [5] for tables). Mathematical formulas can be used in the text (soft elements) or in a separated context (hard elements). We ignore these kinds of tags in the first experimentations described in this article.

2.2.2 Reading Context

Furthermore, we think that this classification introduces a notion that we could call "reading context". A *reading context* is a small part of text, syntactically and semantically self-sufficient, that a person can read in a go, without any interruption. A paragraph or a list item (hard elements) change reading contexts. A footnote (jump element) is inserted into an existing reading context and composes a new one. On the other hand, a bold text (soft element) lies within the current reading context and does not interrupt it.
This definition and this designation are close but not conceptually equivalent to the "context search", suggested by the authors in [8]. The latter only concerns pattern matching techniques, while we think that a reading context can be used by any document engineering technology.

3 Determining tag category

Determining the class or the semantics of a tag name (hard/ soft/jump tags [3], but also emphasis tags [17], equivalent tags [1]) is generally done either manually ("intuitively") or automatically by considering tag names and DTD comments [1]. But intuition has its limits (for example the definition of an emphasis tag is not perfectly clear), and tag names and comments are very approximate clues: tag names are rarely "real words" while comments – their presence, their arrangement, their clarity – are too dependant on the person that wrote them. Designing an efficient and robust system using mainly these factors in a general context seems very uncertain.

The method that we propose here frees from this kind of constraints. It does not use the DTD at all, but the features described in section 2, in order to determine a tag class. This is done through a procedure based on a syntactic analysis of text.

3.1 Procedure

Let tn be the tag name that we want to determine the class of; e an element of this name tn; and p_e its parent element (p_e contains e).

3.1.1 Soft tags

Definition: *e is a* soft element *if one can remove its mark-up and obtain a syntactically correct text in p_e.*

The application of this definition to our examples of section 2.1 leads to the following texts. We can see that asterisked sentences do not mean anything. Therefore, only `'bold'`, `'italics'`, `'underlined'` and `'sc'` are soft tags (cases 3').

(2') * A new study about evolution of tourism in the United States Elections in Ukraine: the Central Commission has published the official results.

(3') a'. United States elections are administered at the state and local levels.

b'. Noam Chomski's comments about United States elections.

c'. United States Elections.

(4') a'. * I heard the news today about United States eleca door snapstions.

b'. * The 2004 United StatesSee an article about the United States of America on page 142 elections caused less controversy than in 2000.

c'. * This document deals with 1995 and 2002 Jacques Chirac J. Chirac, the french president, is, by the way, ...

3.1.2 Jump tags

Definition: *e is a* jump element *if one can remove the entire element (tags + content) and obtain a syntactically correct text in p_e.*

Applied to the running examples, this definition shows that `'comment'` and `'footnote'` are "jump" elements (cases 4").

(3") a". * are administered at the state and local levels.

b". * Noam Chomski's comments about

c". * U S E.

(4") a". I heard the news today about United States elections.

b". The 2004 United States elections caused less controversy than in 2000.

c". This document deals with 1995 and 2002 Jacques elections.

3.1.3 Hard tags

A hard tag has some specific features. However the safest and simplest definition is:
Definition: *a* hard tag *is neither a soft tag nor a jump tag.*

3.2 Comments

Obviously these definitions and procedures cannot have an isolated application, and we cannot consider separately each element of the corpus, for the following reasons:

- If the parent element p_e is not mixed-content (*i.e.* if it does not contain any textual data, but only other XML elements), it will be impossible to conclude, as shows the following example for `'italics'`:

  (5) <title><italics>Space technologies</italics>
  </title>

- If the selected portion contains other elements than some text and tags named tn, it is impossible to reconstitute the real text for certain. Indeed we do not know the class of the other tags.

  (6) <title><bold>Human travel</bold> to
  <italics>Mars</italics> could happen sooner
  than expected.</title>

  In this example, while studying the tag `'italics'`, we do not know whether `'bold'` is a soft, hard or jump tag. Thus we ignore which part of text should be kept. It would be necessary to analyze the corpus several times as we acquire knowledge about the other tags (in the example, re-analyze the portion after having learned that `'bold'` is a soft tag).

- In some cases the described algorithms can lead to the recognition of the same element as "soft" element and as "jump" element:

  (7) Napoléon Bonaparte<footnote>who was born in Corsica in 1769</footnote> died at 52 after having drastically changed the world.

  (8) The Professor <bold>Stephen Hawkins</bold> works on the laws which govern the universe.

  Besides, example 3.c" can be considered as a correct acronymic title.

But we want a tag class to be attributed to a **tag name**, and not to each occurrence of this tag. Hence, we do not need a 100 % precision, but only statistically significant results, *i.e.* allowing to associate a tag name with its category with no doubt. For this reason, we need to have a corpus which is large enough (see some comments and experiments about the size of the corpus in section 4.5).

4 Experimentations

The definitions that are introduced in the previous section allow to perform an automatic classification in certain cases. We performed our experimentations on the INEX [6] collection, which consists of about 12000 scientific articles from various IEEE journals. The structure of this corpus represents both logical structuring, like *sections*, *paragraphs*, *titles*, and presentation tags. The DTD describes 192 different content models.

4.1 Syntactic analysis

A syntactic linguistic analysis is performed in two step:

- A *part-of-speech (POS) tagging*, or the recognition of the grammatical category of each word in a text.

- The application of grammatical rules.

Wa carried out the first part with the free tool TreeTagger [14, 13]. Example 9 presents a POS tagging (with DT = determiner, JJ = adjective, NN and NNS = noun, IN = preposition, PN and PNS = proper noun).

(9) A_{DT} new_{JJ} example_{NN} about_{IN} United_{PN} States_{PNS} elections_{NNS}

The analysis is performed with a set of context-free rules describing some grammatical constructions. As an example, figure 1 lists the rules that will be triggered when parsing our sample phrase 9. A recursive application of these rules results in the *syntactic tree* represented in figure 2[1]. A complete list of rules is given in appendix.

1. *NP → DET? JJ? (NN | NNS)*
2. *NP → NP PP*
3. *NP → NP (NN | NNS)*
4. *NP → (PN | PNS)+*
5. *PP → IN NP*

Figure 1: Examples of context-free rules, with NP = Noun Phrase and PP = Prepositional Phrase. Question mark "?" means that the element is optional in the sequence, the sign "+" that it is repeatable, and "|" represents a logical OR.

4.2 Algorithm

4.2.1 Soft tags

A soft tag is recognized as follows:
For each element e (with tag name tn and parent p_e):

1. We select a portion of text surrounding e. If end-of-sentence markers (strong ponctuation) occur in this text, we reduce this portion to the sentence containing the element:

   (10) <news>

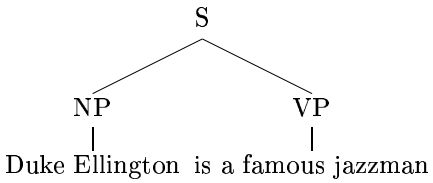
   Sports – Women's Tennis: the Belgian
   <italics>Kim Clijsters</italics> is expected to miss Australian open. She continues to recover from a wrist injury.

   </news>
   → The Belgian <italics>Kim Clijsters</italics> is expected to miss Australian open.

2. If the tag corresponds to one of the configurations described in section 3.2 (non mixed-content parent or other elements), we skip the tag.

3. We remove mark-up from the remaining text and perform a syntactic analysis. If the text still "means" something, *i.e.* if a same syntactic tree groups together the content of the element e and some surrounding words together, then e is a soft element:

   (11) <italics>Duke Ellington</italics> is a famous jazzman.



[1]Note that with a set of rules, different parsings can be obtained. This is a major issue in NLP, but in our case the aim is not to obtain a semantically correct interpretation, but only to prove that the parsing has a solution.

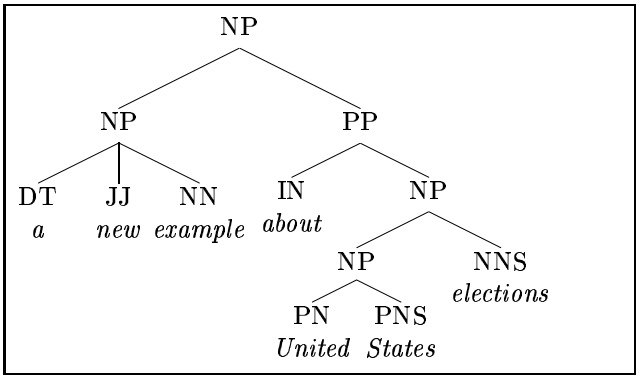


Figure 2: Syntactic tree of sentence 9, obtained with rules of figure 1.

4.2.2 Jump tags

For each element e (with tag name tn and parent p_e):

1. We remove the element e.
2. As for soft tags (and with the same restrictions), we select a portion of text surrounding the point where e occurred; Another constraint is that e should have textual data on its left *and* on its right, contrary to soft tag procedure.
3. We perform a syntactic analysis of the text. If the text still "means" something without e, *i.e.* if a same syntactic tree groups together some words on both sides of the location of e, then e is a jump element.

4.2.3 Hard tags

Any tag not recognized as "soft" or "jump" tag is considered to be "hard".

4.3 Preliminary comments

4.3.1 Terminology

In this section the following terms are used:

- The term **filtering** represents the step 2 of both soft and jump algorithms: the omission of certain tags, because an analysis would not permit to conclude.
- A **successful analysis** is an analysis that meets the conditions of step 3 of both algorithms: syntactic connection between element content and surrounding text for soft algorithm, and syntactic connection between words around the element for jump algorithm.

4.3.2 Linguistic analysis

Automatic parsing of a text is far from being a solved problem for common language. Erroneous analyses, careful though the grammar was designed, will stay very numerous.

Most of linguistic constructions that compose *soft* elements, as well as words preceding *jump* elements, are noun phrases (NP). For this reason, we put a stress on recognition of NPs and of the simplest constructions containing NPs (prepositional phrases, verbal phrases...). We left aside more complex constructions such as relative propositions or interrogative forms.

Setting too many rules would be risking analyzing incorrect phrases. This is no the case in our system, and we can be pretty sure that each syntactic tree is correct; on the other hand, many structures can be missed.
Furthermore, in scientific publications, many tags contain abbreviated expressions or mathematical notations. Here are three examples of very frequent expressions:

(12) a. The Timetables of Technology (rev.), <it>No. 4, 83</it>.

b. The noise source <bu>v(t)</bu> also represents the influence of...

c. For each Y such that <ss>symptom</ss>(X, Y) appears...

These phrases are not recognized by our syntactic rules and are even often misinterpreted by POS tagging.

For these reasons, when we comment the results, we consider that scores higher than 40 or 50 % are very significant results. Indeed they should be compared to the rate of incorrect phrases that are likely to be analyzed as correct by the system. This rate is expected to be very low (< 5 %).

4.3.3 Hard tags vs. the rest of the world

We stated that an element had to be the child of a mixed-content element (*i.e.* to have some textual data around it) to be analyzed. For many tag names, there is no occurrence in the corpus that respects this rule. In accordance with the definitions, these tag names are therefore immediately put into the "hard tags" category. This is the case of all logical tags, like `'bdy'` (body), `'sec'` (sections), `'p'` (paragraphs), `'bib'` (bibliography), `'bm'` (back matter), but also `'au'` (author), `'atl'` (title), `'abs'` (abstract) and elements like list items or table cells. In the set of names eliminated by this way, we did not find any that deserved to go further through our algorithm.
Actually it turns out that this simple separation between elements that have textual data around them and those that have not, is enough to distinguish hard tags from others. This is a quite simple and intuitive definition; but the INEX corpus is particularly well and strongly structured, and some other document sets could not be treated with such a strict partition; this separation should not be considered as a general feature.

4.4 General Results

The following notations are used in our figures:

- n = total number of occurrences of a tag name in the corpus;
- p_f = percentage of these occurrences that is kept after filtering, for soft or jump algorithms (see definitions in section 3.1);
- s_f = percentage of these occurrences (kept after filtering) that are successfully analyzed, for soft or jump algorithms;
- s_t = percentage of all occurrences (before filtering) that are successfully analyzed ($s_t = s_f \times p_f$).

Considering the large number of different tag names in the studied corpus, it would be long and useless to list the results for each one. As we said, structural elements like paragraphs, sections, lists, tables, figures, and data such as authors or titles did not overcome the filtering ($p_f = 0$) and have already been classified as hard tags.

4.4.1 Soft tags

Table 1 gives some results for the "soft tag" algorithm. With tag names, between parentheses, is the "semantics" given by comments in the DTD. The most important column is the bold one (s_f), giving the rate of treated elements that correspond to the definition (according to our system).

Tag name	n	p_f	s_f	s_t
a (link)	91	60.44 %	**74.55 %**	45.05 %
ariel	5800	42.38 %	**67.98 %**	28.81 %
b (bold)	160171	38.03 %	**61.40 %**	23.35 %
bi (emphasis)	4233	12.93 %	**58.32 %**	7.54 %
bu (emphasis)	206	29.61 %	**31.15 %**	9.22 %
bui (emphasis)	53	13.20 %	**42.86 %**	13.21 %
it (italics)	1070877	32.94 %	**60.46 %**	19.92 %
large	1	100.00 %	**100.00 %**	100.00 %
math	76270	61.06 %	**17.84 %**	10.89 %
ref (hyperlink)	395946	79.21 %	**2.86 %**	2.27 %
rm (roman)	3548	42.45 %	**49.60 %**	21.05 %
scp (small caps)	114255	52.62 %	**72.70 %**	49.22 %
ss (typeface)	4402	62.49 %	**50.97 %**	31.85 %
sub (subscript)	291661	19.64 %	**21.80 %**	4.28 %
super	44567	31.18 %	**28.93 %**	9.02 %
tt (typeface)	47517	64.45 %	**59.72 %**	38.45 %
u (underlined)	2713	30.48 %	**37.97 %**	11.57 %
ub (med. bold)	2	50.00 %	**100.00 %**	50.00 %
url	25050	54.32 %	**48.39 %**	26.28 %

Table 1: Examples of soft tag searching experiments.

For emphasis tags[2], most of the scores are between 40 and 70 %. For two of them (`'large'` and `'ub'`), because of the very small number of occurrences, the results are not significant (even if we can notice that we have each time *good* unsignificant results).

The `'math'` element introduces a mathematical environment, and `'super'` and `'sub'` are only used in this context. The result for these tag names is not as clear as for the others; we already noticed in section 2 that it could be considered as an exception in our classification.

Interesting results are those obtained for `'a'`, `'url'` and `'ref'`. The first two tag names represent quite the same information, that is links to URLs. The difference is that `'a'` has an `'href'` attribute and can then contain some text while `'url'` contains only the URL. Both tags have success rates that are comparable with emphasis elements. And indeed a look at the documents shows that these tags are generally used without changing the reading context:

(13) a. Additional information can be found at
<url>http://...</url>.

b. The complete survey results showed that ...

Finally, `'ref'` groups together hyperlinks to citations, figures, footnotes and other elements. Its success rate for "soft" evaluation is close to zero. And indeed uses of `'ref'` elements are typical of jump tags.

[2]In the whole article, we call *"emphasis tag"* any *presentational* or *procedural* markup (see [4]). All elements that are concerned with the presentation of the text, independently of its structure, go into this category. We do not make the distinction between physical and logical presentation markup. Thus, for example, HTML tags `'EM'` and `'STRONG'` (*logical* tags, because they precise a *role*) are considered to be emphasis tags, as well as `'B'` or `'I'` (*physical* tags, because they directly describe an *appearance*). On the other hand, `'TITLE'`, `'H1'` or `'P'` concern the structure of the document, and thus are not emphasis tags.

Tag name	n	p_f	s_f	s_t
a (link)	91	48.35 %	**48.84 %**	23.08 %
b (bold)	160171	9.88 %	**24.22 %**	2.39 %
it (italics)	1070877	25.88 %	**21.83 %**	5.65 %
math	76270	52.75 %	**47.68 %**	25.15 %
ref (hyperlink)	395946	59.57 %	**35.86 %**	21.26 %
ss (typeface)	4402	57.97 %	**22.41 %**	12.99 %
url	25050	26.14 %	**22.43 %**	5.86 %

Table 2: Examples of jump tag searching experiments.

4.4.2 Jump tags

Table 2 contains some examples of results obtained with the "jump tag" procedure. We only kept `'b'`, `'it'` and `'ss'` as representants of emphasis tags, because all results for this category are similar. The value p_f is always lower than in "soft tag" experiments because we have more constraints.
The figures are, in this case, less clear-cut than previous ones. All names that had a good score in soft searching get around 20 % here. This is much lower but not negligible. For `'ref'`, we found 36 % of successful analyses. The difference is big when compared with its score for the soft algorithm; but with the same algorithm, other elements are not much lower. `'math'` still gets strange results, and `'a'` score is much higher than expected (the number of non-filtered occurrences may be too low for this tag name).

4.4.3 Comments and final results

The results for soft tag searching are very clear and good. And this is pretty good news, because making the distinction between soft tags and the others leads to one of the most promising of the envisaged applications (see section 5). The "jump" results are also good but less clear. That is why we propose the following adapted procedure for each tag name tn:

1. If the soft score is higher than a given threshold, then tn is a soft tag name.
2. **If not**, and only if not, the procedure for detecting jump tags is used. If the jump score is higher than a (possibly different) threshold, then tn is a jump tag name.
3. Else it is a hard tag name.

As no element from our corpus reached the third step, we cannot prove that a non-filtered hard tag would not be taken as a soft or jump tag. But hard tags are very particular; their text often ends by a strong ponctuation (paragraphs, list items...), or is not natural language at all (as ISBN or volume number). In both cases a sentence-based syntactic analysis would not work. An application to another set of documents (not as well-structured) should be envisaged to confirm this idea.

If we set thresholds to 20 % (which is an approximate median between 0 and the lowest "soft" scores), we obtain the following final results. The terms used in this list are composed of DTD comments and entity declarations:

- Soft tags:
  - tface: `'tt'` (typewriter font) and `'ss'` (sans serif)
  - fweight: `'b'` (bold) and `'ub'` (medium)
  - fslant: `'it'` (italics) and `'rm'` (roman)
  - fspec: `'scp'` (small caps)
  - fattr: `'u'` (underlined)

- tpos: `'sub'` (subscript) and `'super'` (superscript)
- tsize: `'large'`
- emph: `'ariel'`, `'bi'`, `'bu'`, `'bui'`
- ref: `'a'` and `'url'` (link to url)

- Jump tags:
  - ref: `'ref'` (hyperlink)
  - `'math'`

- Hard tags:
  - all the others...

We are conscious that this experimentation should be performed on a new set of documents in order to confirm our threshold setting and our entire procedure, as well as the tests on corpus size described in the next section. Unfortunately we do not have access to any other large document-centric XML corpus.

4.4.4 Scope

This automatic experimentation does not apply to all sets of XML documents, and not even to all document-centric XML documents. In particular an important constraint is that the considered corpus must have an appriopriate size (but this minimum size is not so big – see section 4.5). Anyway we think that the soft/jump/hard tag types and their precise definition given in section 3 have real usefulness and applications (see section 5). This utility does not depend on the fact that the categorization is automatic or not in practice. Nevertheless we think that, when it works, such an automatic process can be very useful.

4.5 Corpus size

An important question is the size of the set of documents (sharing the same DTD) that we need in order to get relevant results from our algorithms. More precisely, the key value is the number of occurrences of each tag name. INEX corpus is much bigger than necessary in most cases, but some tag names still appear only a few times (`'large'`, `'ub'`, `'bui'`, `'a'`...).

For each tag name, we split INEX corpus into subparts containing a given equal number o of tag occurrences. We also did this partition with the corpus after filtering (see 3.1).
We ran our system on these different corpus fractions. The results of these new experiments give a good idea of the requisite amount of data[3].

For a given number o and a tag name tn, we obtain n subparts containing o occurrences of tags tn. We are interested in the percentages of these subparts for which the analysis leads to the same conclusion than for the entire collection.
Figures 3 and 4 present these rates for five representative tag names, with o varying from 1 to 2000 (n is function of o and tn). In the legend, after tag names, is recalled the total number of occurrences in the whole corpus.
Figure 3 is the most important; it represents the results for non-filtered sets. Unsurprisingly, rates for the filtered collection (figure 4) are much better; in the case of small corpora an effort on filtering could be effective (multiple parsings for example – see comments in section 3.1.1).
For example, one can read that when the collection is divided into subcollections containing 200 `'b'` tags (without filtering), about 80 % of these subcollections, analyzed separately, lead to the conclusion that `'b'` is a soft tag name.

[3]The subcorpora do not have the same size but contain the same number of tags of one kind. We could have divided the corpus into subparts of same size; but we think that this would have given a more corpus-dependant view on influence of size upon the results. The number of occurrences is a general value, applicable to any collection.

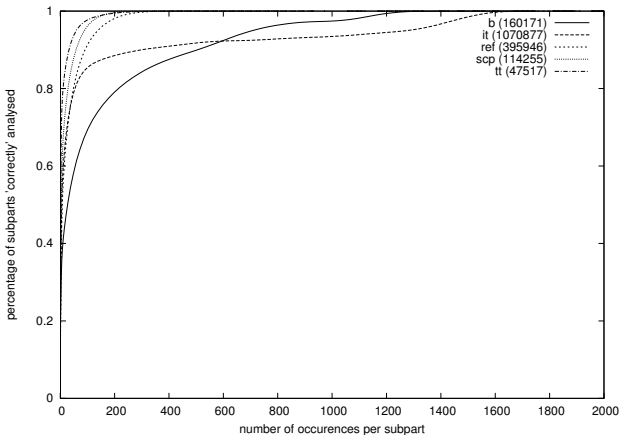


Figure 3: Corpus size experiments on non filtered tags.

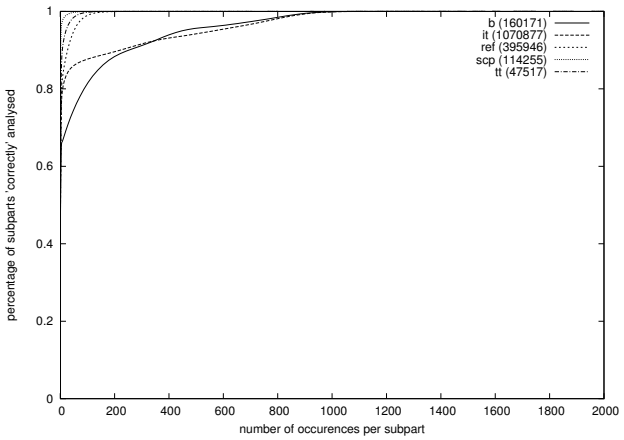


Figure 4: Corpus size experiments on filtered tags.

Tag names `'it'` and `'b'` are those that get the "worst" results: more than 1000 or 1500 tags are necessary to obtain a trustworthy result. For all the other tags (included tags that are not represented here), a number between 100 and 300 occurrences is enough to conclude in each case.
The reason is that `'it'` and `'b'` are often used in INEX collection for indexes or bibliographic information. Many tags in such elements are either removed by filtering (for example, title texts in indexes are entirely `bold`, and not mixed-content) or very hard to analyze, because they contain some abbreviations (journal names, dates, series number).
As these elements are heterogeneously spread over the corpus, some subparts which are totally or partly composed of an index will get much lower rates than wanted. A better set of grammar rules might improve the results in this case. But `'it'` and `'b'` are also much more commonly used than other tag names, and therefore a appropriate number of occurrences is reached very soon in the collection.

Retrospectively these experiments on corpus size show that results concerning rare tags like `'a'` and `'bui'` were probably not significant, even if not surprising.

5 Comments and perspectives

Soft/jump/hard classification of XML tag names, either automatically obtained or not, opens up the way to many uses; The following are some trails that we intend to follow in our further works. Aside from the first one, they are all based on our new concept of "reading context":

- In almost all cases, soft tags are ***emphasis tags*** (bold, italics, underlined... See footnote 2). These tags are generally used to express the importance of some words or phrases in the text[4]. INEX corpus shows that a DTD can define many tags of this kind; their names can be diverse and not very intuitive. However, this information is quite important, and in Information Retrieval especially, where many systems use emphasis tags to give more weight to the terms that they contain. So far, emphasis tags were often manually collected, or sometimes defined as "any child of a mixed-content element" [17] (but this definition is too general; it includes at least all soft and jump tags – see 4.3.3. Moreover it handles individual occurrences rather than types).

- An indexing issue can be solved, that is the way of dealing with tags during the document indexing. In the following example 14.a, indexed words should be *'Tom'* and *'Sawyer'* in spite of `'sc'` tags (small capitals) cutting them. In this case we need to consider these tags as transparent.
  But if we apply this method to examples 14.b and 14.c, then *'MarkTwain'* and *'1876The'* are indexed as unique terms, while we should obviously separate words *'Mark'* and *'Twain'*, *'1876'* and *'The'*. Here `'fn'` and `'ln'` tags must be replaced by blank characters.
  Finally *'Clemens'* is a single term in 14.d, and the `'correction'` element need to be analysed separately.

  Knowing the jump/soft/hard features of each tag name allows these distinctions (`'sc'` is *soft* while `'note'` and `'correction'` are *jump*).

```
(14) a. <title>
            T<sc>om</sc> S<sc>awyer</sc>
        </title>
     b. <author>
            <fn>Mark</fn><ln>Twain</ln>
        </author>
     c. Book written in 1876<note>The author was born in 1835.</note>.
     d. <transcription>
            His real name was Samuel Langhorne
            Clem<correction>the first transcriptor
            wrote a double 'm' here</correction>ens.
        </transcription>
```

- This categorization allows to distinguish *physical proximity*, in the XML file, from what we could call ***logical proximity***. Logical proximity depends on the arrangement of the terms in the structure. Again it is closely related to the concept of "reading context". Two words separated by a start/end soft tag are *logically* adjacent because they are consecutive in the same reading context. This is not the case when mark-up represents jump or hard tags, because the words can belong to different reading contexts.
  This is particularly interesting in Information Retrieval when the query is composed of several words (either distinct terms or "multi-word terms" – as *"information retrieval"*).
  Suppose that one is looking for information about U.S. elections. The set of examples proposed in section 2 proves that the physical proximity of terms "United States" and "Elections" is

[4]Exceptions in our corpus are `'a'` and `'url'`, which are soft tags but not emphasis tags, but they do not contain real words. In [8], another counterexample is given, called "integration", as in *"this is an im<correction>m</correction>ediate reaction"*.

not a guarantee of relevance. Relevance should rather be related to logical proximity. Thus examples 3.a, 3.b, 4.a and 4.b are relevant, while 2 and 4.c are not.

Note that this definition of proximity is also applicable to lists, and this is particularly interesting. Lists are composed of an introduction, one or more items and sometimes a conclusion [2]. In this context, there is the same logical proximity between the introduction and each item, and between each item and the conclusion, regardless of the number and the length of the items.

(15) Macro-nutrients are constituted by:
<list>
<item>carbohydrates: general molecular formula CH_2O ...</item>
<item>lipids, all hydrophobic, ...</item>
<item>proteins, constructed with
amino-acids...</item>
</list>
and are the basic material from which the body is built.

In this example, the words *"lipids"* and *"proteins"* are *physically* far from the term *"macro-nutrients"*, but *logically* equally close to it.

- Document-centric semi-structured documents are, as well as flat (non structured) documents, a good playing field for natural language processing researchers. But in the case of XML, an additional issue is the necessity and the difficulty to preserve reading contexts (what a human actually read). Yet this is the condition for performing a correct part-of-speech tagging, which is often the first step of a NLP work, but also for any kind of advanced syntactic/semantic linguistic analyses. Our set of examples can be read "in reverse" to illustrate this problem (especially number 14 for POS tagging and 2, 3 and 4 for syntactic analyses): an appropriate treatment of tags is necessary to recover the reading context and to analyze the text properly.

  We saw that knowing the class of each element in the corpus solves by definition this problem, as our interpretation of soft/jump/hard classes is a direct answer to this point.

6 Conclusion

In this article, we went back over a simple division between soft, jump and hard tags. We presented a definition of a new concept of "reading contexts" that uses these tag types, and that fits with the way human people read a structured document. This new way to explore XML documents should come in useful for many document engineering applications.

We also proposed a simple method for determining tag classes of a corpus automatically in certain cases, with only a few exceptions, like lists, tables or mathematical environments. Further works that should be accomplished in this side are: an extension to all exceptions, and especially lists; an application to other corpora; and a more accurate procedure, in terms of filtering and of syntactic parsing, in order to analyze as many tags as possible properly, and then to get good results for smaller corpora.

The definition of a new concept of "logical proximity" using reading contexts should help multi-term retrieval in textual content and in lists. The recognition of emphasis tags allows to give appropriate weights to more important terms. Soft/jump/hard distinction and reading contexts also helps POS tagging and other natural language analyses of XML corpora. And we have no doubt that other uses can be found.

References

[1] S. Abiteboul, I. Manolescu, B. Nguyen, and N. Preda. A Test Platform for the INEX Heterogeneous Track. In Fuhr et al. [6].

[2] S. Aït-Mokhtar, V. Lux, and E. Banik. Linguistic Parsing of Lists in Structured Documents.

[3] D. Colazzo, C. Sartiani, A. Albano, G. Ghelli, P. Manghi, L. Lini, and M. Paoli. A Typed Text Retrieval Query Language for XML Documents. *Journal of American Society for Information Science and Technology (JASIST), Special Topic Issue on XML and Information Retrieval*, 53(6):467–488, Apr. 2002.

[4] J. H. Coombs, A. H. Renear, and S. J. DeRose. Markup Systems and the Future of Scholarly Text Processing. *Communications of the Association for Computing Machinery*, 30(11):933–947, 1987.

[5] S. Douglas, M. Hurst, and D. Quinn. Using Natural Language Processing for Identifying and Interpreting Tables in Plain Text. In *Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval*, pages 535–546, Las Vegas, Nevada, USA, 1995.

[6] N. Fuhr, M. Lalmas, S. Malik, and Z. Szlàvik, editors. *Advances in XML Information Retrieval. Third Workshop of the Initiative for the Evaluation of XML retrieval (INEX)*, volume 1299 of *Lecture Notes in Computer Science*, Schloss Dagstuhl, Germany, Dec. 2004. Springer-Verlag.

[7] C. F. Goldfarb. *The SGML Handbook*. Oxford University Press, 1991.

[8] L. Lini, D. Lombardini, M. Paoli, D. Colazzo, and C. Sartiani. XTReSy: A Text Retrieval System for XML documents. In D. Buzzetti, H. Short, and G. Pancalddella, editors, *Augmenting Comprehension: Digital Tools for the History of Ideas*. Office for Humanities Communication Publications, King's College, London, 2001.

[9] B. S. Mitchell P. Marcus and M. A. Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics (Special Issue on Using Large Corpora)*, 19(2):313–330, June 1993.

[10] W. Piez. Beyond the "descriptive vs. procedural" distinction. *Markup Languages: Theory and Practice*, 3(2):141–172, Dec. 2001.

[11] A. Renear. The descriptive/procedural distinction is flawed. *Markup Languages: Theory and Practice*, 2(4):411–420, 2000.

[12] A. Renear, D. Dubin, C. M. Sperberg-McQueen, and C. Huitfeldt. Towards a Semantics for XML Markup. In *Proceedings of the 2002 ACM Symposium on Document Engineering*, pages 119–126, McLean, Virginia, USA, 2002. ACMPress.

[13] TreeTagger - a language independent part-of-speech tagger. http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/ DecisionTreeTagger.html.

[14] H. Schmid. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *International Conference on New Methods in Language Processing*, Sept. 1994.

[15] I. Song and P. S. Bayerl. Semantics of XML documents. Internal Working Paper, Apr. 2003.

[16] C. M. Sperberg-Mcqueen, C. Huitfeldt, and A. Renear. Meaning and interpretation of markup. *Markup Languages: Theory and Practice*, 2(3):215–234, Aug. 2000.

[17] R. van Zwol, F. Wiering, and V. Dignum. The Utrecht Blend: Basic Ingredients for an XML Retrieval System. In Fuhr et al. [6].

[18] Extensible Markup Language (XML). World Wide Web Consortium (W3C) Recommandation. http://www.w3.org/TR/REC-xml/.

[19] Semantic Web. World Wide Web Consortium (W3C). http://www.w3.org/2001/sw/.

[20] XQuery 1.0: An XML Query Language. World Wide Web Consortium (W3C) Working Draft. http://www.w3.org/XML/Schema.

A Rules

Here are the most of the rules used for our experimentations. A few semantic actions are associated with some rules, in order to avoid some erroneous analyses, but we do not dwell on that. Rules are all context-free rules. Elements that have ***this font*** are POS element. Their names correspond to the Penn Treebank Tag Set [9].

q → *ng ivg*
q → ***wp*** *ivg*
q → ***wrb*** *ivg*
s → *ng vg*
s → *vg(imp)*

nc → *"of" ng*
pp → ***in*** *ng*
pp → ***in*** *date*
relProp → ***wdt/wrb*** *s*
relProp → ***wdt/wp*** *vg*

ng → *"(" ng ")"*
ng → *np*
ng → *ng sep ng*
ng → *ng relProp*
ng → *ng vg(pres. part.)*
ng → *ng vg(past part.)*
ng → *ng nc*
ng → *ng pp*
ng → *ng* ***jj***
ng → *np np(quot)*
ng → ***jj***

np → ***dt nn***
np → ***nn***
np → ***pn***+
np → *quot*
np → ***nn***+

vg → *v*
vg → *vg ng*
vg → *vg pp*
vg → *sep vg*
vg → *vg(passive) "by" ng*
vg → *vg(past part.) "by" ng*

ivg → *ng vg*
ivg → *aux ng vg*
ivg → *vg ng*

date → *year*
date → *"year" year*
date → *year* ***pun*** *year*
date → ***in*** *year*
year → ***cd***
quot → " * "
sep → ***pun***
sep → ***cc***

Les rapports de recherche
du Centre G2I de l'ENSM-SE
sont disponibles en format PDF
sur le site Web de l'Ecole

G2I research reports
are available in PDF format
on the site Web of ENSM-SE

www.emse.fr

Centre G2I
Génie Industriel et Informatique

Division for
Industrial Engineering and Computer Sciences
(G2I)

Par courier :

By mail:

Ecole Nationale Supérieure des Mines de Saint-Etienne
Centre G2I
158, Cours Fauriel
42023 SAINT-ETIENNE CEDEX 2
France

Ecole Nationale Supérieure des Mines de Saint-Etienne
Centre G2I
158, Cours Fauriel
42023 SAINT-ETIENNE CEDEX 2

www.emse.fr